2008 IEEE/RSJ International Conference on Intelligent Robots and Systems
Acropolis Convention Center
Nice, France, Sept, 22-26, 2008

Active Sensing based Dynamical Object Feature Extraction

Shun Nishide, Tetsuya Ogata, Ryunosuke Yokoya, Jun Tani, Kazunori Komatani, and Hiroshi G. Okuno

Abstract— **This paper presents a method to autonomously extract object features that describe their dynamics from active sensing experiences. The model is composed of a dynamics learning module and a feature extraction module. Recurrent Neural Network with Parametric Bias (RNNPB) is utilized for the dynamics learning module, learning and self-organizing the sequences of robot and object motions. A hierarchical neural network is linked to the input of RNNPB as the feature extraction module for extracting object features that describe the object motions. The two modules are simultaneously trained using image and motion sequences acquired from the robot's active sensing with objects. Experiments are performed with the robot's pushing motion with a variety of objects to generate sliding, falling over, bouncing, and rolling motions. The results have shown that the model is capable of extracting features that distinguish the characteristics of object dynamics.**

I. INTRODUCTION

Recent studies on affordance-based motion [1] generation have provided a foresight into creation of intelligent robots. Affordance is a feature of an object or environment that implies how to interact with the object or environment. Works on affordance-based robot navigation focus on predicting the traversability of the environment [2] [3]. Robots used in these studies acquire knowledge of the relation between its own behavior and resulting object dynamics based on active sensing [4] for evaluating traversability of the current environment. Works on affordance-based object manipulation also adopt active sensing to generate motions based on a criteria. Some examples of these criteria are extension of reach [5] or reliable predictability [6]. Although these works have succeeded in generating appropriate motions, they all predefine a challenge required for environment adaptive robots; Features for describing the objects are predefined. Therefore, these works were capable of generating motions for objects with similar shapes as those used for training. By providing robots the ability to autonomously extract object features, the robot will be able to select the appropriate features based on the given task. The work could be combined with previous studies [2] [3] [5] [6] to generate motions based on a given criteria.

Related works on vision grounding have been done by Fitzpatrick and Metta [7] [8]. These works focus on segmenting objects from visual images based on the robots active sensing experiences with objects. The knowledge acquired during these experiences would be used to learn the objects affordances and be applied to generate mimicry motions. While these works provide the robot the knowledge to segment objects from motion, our method provides the robot the knowledge of how the object motion is to be represented.

S. Nishide, T. Ogata, R. Yokoya, K. Kazunori, and H. G. Okuno are with the Department of Intelligence Science and Technology, Graduate School of Informatics, Kyoto University, Kyoto, Japan {nishide, ogata, yokoya, komatani, okuno}@kuis.kyoto-u.ac.jp

J. Tani is with the Brain Science Institute, RIKEN, Saitama, Japan tani@brain.riken.jp

Based on affordance theory, two types of features, or invariants, exist considering perception of environments [9]. Structural invariants are features that describe the environment itself. These features remain unchanged when the environment is changed by an event. Transformational invariants are features that describe the change of the environment. The same pattern of transformational invariants occurs for same events. An example of these invariants are, the shape or texture of an object, which are structural invariants as they do not change when the object is pushed, and the transition of the center position of an object, which is a transformational invariant as it describes how the object is moving. In this paper, the authors use the term "static features" for structural invariants, and "dynamic features" for transformational invariants as these terms are more intuitive. The term "invariant" is used to relate the work with affordance theory.

The objective of this research is to autonomously extract static and dynamic features based on active sensing, for applying to motion generation. The authors have developed a model that extracts static object features from background eliminated raw images [10]. The work trained and self-organized object dynamics, described by center position and inclination of the principal axis of inertia of the object image, and linking them with background eliminated raw object images. As a result, the model self-extracted static object features representing fineness, roundness, and sharpness of the objects. These static features greatly affect the rolling, falling over, sliding motions of the objects used in the experiment. However, as the dynamic features were predefined in the experiment, the method was incapable of self-organizing a larger variety of objects dynamics.

In this paper, the authors report a technique to autonomously extract dynamic object features. The work contains two issues:

1) Construction of dynamic object feature extraction model.
2) Developing a model training method.

To deal with the first issue, the authors coupled a dynamics learning module with a feature extraction module to extract dynamic object features from the output error of the dynamics learning module. The authors utilize Recurrent Neural Network (RNN) for the robot-object dynamics learning module. There are two reasons for this selection. First, the hardware limitations of the robot (moving the robot too much would lead to damage of hardware) calls for adaptation to unknown environments from few training data. The generalization capability of RNN is one of the capabilities that meet this requirement. Second, we aim to train the dynamics learning module and feature extraction module reciprocally, as independent training of the two modules would not derive the features that describe the dynamics. Neural network is the only model possessing both functions required for realizing the system. To deal with the second issue, the authors have simultaneously trained the two models. This method was decided empirically from comparison with training the two models alternately. Alternate training fixes one model while training the other, alternating the process after several calculation loops. Simultaneous training, on the other hand, trains both models every calculation loop. Since the model requires reciprocal training, simultaneous training is more suitable for the method as it mutually minimizes the errors of the two models. The results have shown that the method is capable of extracting object features that describe the dynamics of the objects.

The rest of the paper is composed as follows. Section II describes the overview of the technique. Section III describes the experiment environment and conditions. Section IV describes the experimental results. Section V presents some discussions considering the results. Conclusions and future works are presented in Section VI.

II. OVERVIEW OF TECHNIQUE

This section describes the overview of the technique. First, the robot acquires its motor value sequences and raw object image sequences during active sensing with objects. These sequences are used to train the model composed by a dynamics learning module and a feature extraction module. The authors utilize Recurrent Neural Network with Parametric Bias (RNNPB), proposed by Tani [11], for the dynamics learning module, and a hierarchical neural network for the feature extraction module. RNNPB inputs the motor/object feature values of the current state and outputs the motor/object feature values of the next state. The hierarchical neural network inputs the raw object image and outputs the object feature. The output of the hierarchical neural network is linked to the input of RNNPB. The model autonomously extracts object features that describe the dynamics of the object in the output of the hierarchical neural network (input of RNNPB) during the training process of the model. As explained in the previous section, the model requires two networks possessing both the functions of dynamics learning and feature extraction. Neural networks are the only models possessing these functions. The construction of the system is shown in Fig. 1.

A. RNNPB Model

RNNPB, shown in the upper half of Fig. 1, is an extension to the Jordan-type RNN [12] containing Parametric Bias (PB) nodes in the input layer. In order to deal with sequential data (dynamics), RNNPB is set as a predictor which calculates the next state $S(t+1)$ from the current state $S(t)$. The

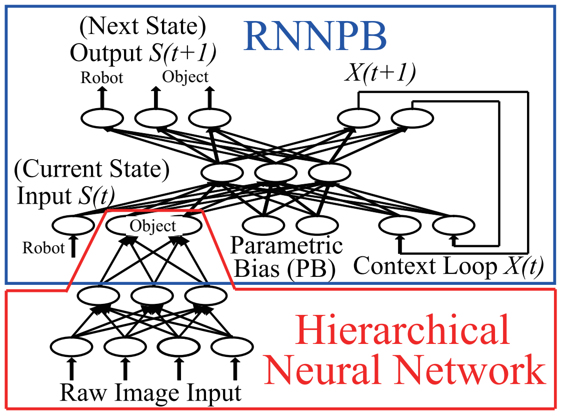


Fig. 1. Construction of the Model

input/output nodes are divided and assigned into nodes that input/output the robot motor values and object feature values.

The role of the PB nodes is to learn multiple sequential data in a single model. While RNN calculates a unique output from the input and context value, RNNPB is capable of altering its output by changing the values of the PB nodes (PB values). In other words, the PB values are used for switching over each sequential data. This capability provides RNNPB to learn and generate multiple sequences. Therefore, RNNPB is often called a distributed representation of multiple RNNs.

RNNPB is a supervised learning system requiring teaching signals as is the Jordan-type RNN. The training phase consists of weight optimization and self-organization of PB values using back propagation through time (BPTT) algorithm [13]. For updating PB values, the back-propagated errors of the weights are accumulated along the sequences. Denoting the step length of a sequence as T, the update equations for PB during the training phase are

$$\Delta\rho = \varepsilon \cdot \sum_{t=1}^{T} \delta_t^{bp} \tag{1}$$

$$p = sigmoid(\rho). \tag{2}$$

First, the delta force $\Delta\rho$ for updating the internal values of PB p is calculated by (1). The delta error δ_t^{bp} in (1) is calculated by back propagating the output errors from the output nodes to the PB nodes. The new PB value p is calculated by (2) applying the sigmoid function to the internal value ρ which is updated using the delta force. ε is a learning constant. The BPTT algorithm updating only the PB values (not updating the weights) is often used for recognition/generation [14].

Training of RNNPB self-organizes the PB values, which encode each sequence, according to their similarities, forming the PB space which creates clusters of similar sequences. The sequences could be reconstructed from the PB values by recursively inputting the output $S(t+1)$ back into the input $S(t)$. This process, called *association*, calculates the whole sequence from an initial state $S(0)$, initial context $X(0)$, and a PB value.

B. Training the Model

The authors use BPTT algorithm for training RNNPB and back propagation (BP) algorithm for training the hierarchical neural network. Considering training for RNNPB, the next state robot motor value is the teaching signal of the robot motor output. Teaching signals for the hierarchical neural network and object feature output of RNNPB are not constant as they are to be extracted autonomously from the training process. Therefore, optimization based on simultaneous training is required for training the model.

Teaching signals for the two neural networks are each acquired from the other neural network. For RNNPB, the output of the hierarchical neural network in the next state $(t+1)$ is used as teaching signals of the object feature output for the current state (t). For the hierarchical neural network, the object feature output of RNNPB of the previous state $(t-1)$ is used as teaching signals for the current state (t). As the two networks are both trained using the output signals from the other network as teaching signals, the two outputs of the networks converge to a uniform value sequence. This value sequence represents both the feature extracted from the image sequence and the value sequence capable of training the RNNPB. Therefore, the training extracts the most appropriate object feature sequence representing the object dynamics.

III. Experimental Setup

The authors used the humanoid robot Robovie-IIs [15] shown in Fig. 2 for evaluation of the method. Robovie-IIs has three DOF (degrees of freedom) on the neck and four DOF on each arm. It also has two CCD cameras on the head for processing visual information, one of which was used in the experiment.

The training procedure of the experiment is as follows.

1) Acquire sequences of images and robot motor values while the robot pushes objects.
2) Extract the object from images by deleting background using color information.
3) Train model using motion sequences.

In this experiment, the authors use background eliminated raw images for simplicity. Evaluating the model for usage with completely raw images is left as future work. The objects used for the experiment are shown in Fig. 3.

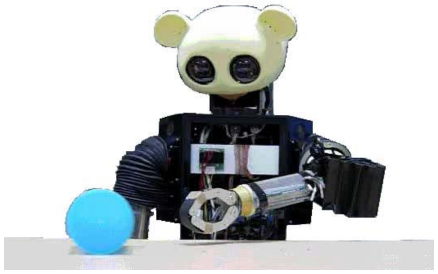


Fig. 2. Humanoid Robot Robovie-IIs

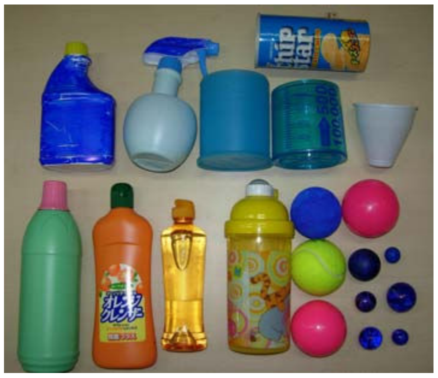


Fig. 3. Objects used for Experiment

A. Motion Sequence Acquisition from Active Sensing

Robovie II-s conducted pushing motion with its left arm for each of the objects shown in Fig. 3. Pushing motions were generated at five different heights by altering Robovie II-s' shoulder pitch angle. As a consequence, the objects generated sliding, falling over, bouncing, and rolling motions. The balls were put on the cup to create bouncing motions when pushed. Excluding the sequences that the robot's arm didn't hit the object and those that the object couldn't be extracted (due to occlusions or illumination conditions), a total of 59 sequences were acquired. The number of each object motion out of the total is shown in Table I. The small number of fall over motions is due to occlusions that occur when the object is hit with a high robot arm motion. As a result, more than half of the whole falling over motions were eliminated for use in the experiment. The sliding motion, on the other hand, is a stable motion having less problems with the object extraction process. Therefore, the total number of sliding motions is larger compared to the other motions.

During the pushing motion of the robot, image and motor sequences were acquired at 10 frames/sec. Acquisition of the sequences were started just before the robot's arm has had contact with the object, and ended after acquiring 10 steps of data. This was decided by the visible area of the robot's camera; some of the objects went out of view after 11 steps. Although this experiment was conducted under a fixed neck condition, the model could also adapt to cases where the robot constantly tracks the object [14]. An example of an object image sequence where the robot pushed an odor eliminating container to make it fall over is shown in Fig. 4.

B. Configuration of the Neural Networks

The configuration of the neural networks (e.g. number of nodes) greatly affects the training result. As the two linked

TABLE I
NUMBER OF EACH OBJECT MOTION

Slide	Fall Over	Bounce	Roll
24	9	13	13

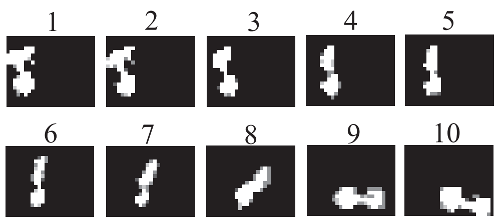


Fig. 4. Image Sequence of an Odor Eliminating Container Falling Over

neural networks each create the teaching signals from the other neural network, this setup is specifically crucial for the training process of the proposed model to converge. The training process would not converge to an optimal solution with a small number of nodes, while a large number of nodes would result in inadequate object features which do not represent the object dynamics, as the model would possess many local minimums. In this paper, the authors focus on the capability of the model to automatically extract dynamical features and self-organize their similarities. Therefore, the configuration of the neural networks were decided empirically. The model could be extended to gain higher training capabilities by combining with works examining the relation between the PB nodes and the self-organization capability [16].

The configurations of RNNPB and the hierarchical neural network are shown in Table II and Table III, respectively. The initial weights of the neural networks were decided randomly within a value between [-0.7, 0.7]. The total number of input/output nodes in RNNPB are 5; 1 is for the robot shoulder pitch angle normalized to [0,1], and 4 for the dynamic object features to be automatically extracted. The input of the hierarchical neural network consists of the grayscale sequential object image, reduced to the resolution 25×20. The output is linked to the object input nodes of RNNPB. The model was trained by iterating the BPTT and BP calculation one million times.

IV. Experimental Results

The authors evaluated the method based on the clustering results of the PB space. As described before, training RNNPB creates clusters of similar sequences into the PB space. Therefore, formation of clusters for the PB values of the same object dynamics implies that the appropriate dynamical object features have been extracted for describing each dynamics.

A. Self-Organized PB Space

Training the model generates the PB space based on the raw image and motor sequences of the 59 training patterns. The generated PB space is shown in Fig. 5. The rhombus, square, triangle, and x mark, correspond to the PB values of the sliding, falling over, bouncing, and rolling motions, respectively.

B. Analyzing the PB Space

The authors have analyzed the PB space as follows, in order to investigate the clusters of each object dynamics.

1) Divide the PB space into 100×100 segments.
2) For each pattern, calculate the absolute error of the object feature sequence and the *associated* sequence. This error is calculated by accumulating the errors along the whole sequence.
3) For each segment, find the minimum error out of the whole patterns.
4) Normalize the error to image format [0, 255] (minimum error = 255, maximum error = 0).
5) Color each segment based on the dynamics and normalized error.

The result of the analysis is shown in Fig. 6. This analysis of the PB space is not an accurate analysis as a PB value may represent two different dynamics if the initial object posture differs. In this analysis, the authors have neglected patterns with large training errors (subraction of object feature sequence by *associated sequence* for the obtained PB). For this experiment, the authors have excluded patterns with training errors larger than 0.02. 24 patterns were excluded, leaving 35 for analyzing the PB space. The blue, green, red, and yellow areas correspond to areas with the minimal errors of slide, fall over, bounce, and roll motions, respectively. Areas with

TABLE II
Configuration of RNNPB

Number of Motor Input/Output Nodes	1
Number of Object Input/Output Nodes	4
Number of Middle Nodes	20
Number of Context Nodes	20
Number of PB Nodes	2
Learning Constant ε	0.01

TABLE III
Configuration of Hierarchical Neural Network

Number of Input Nodes	500
Number of Middle Nodes	30
Number of Output Nodes	4
Learning Constant ε'	0.1

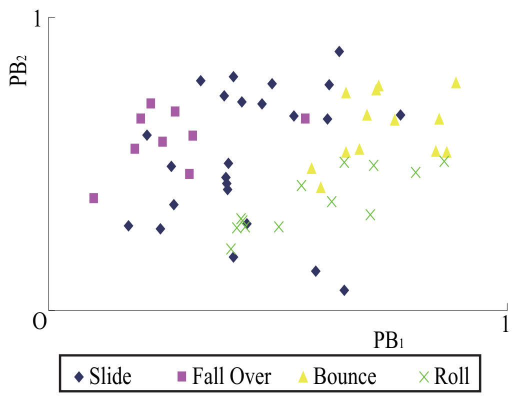


Fig. 5. Generated PB Space

lighter colors have smaller errors, while areas with darker colors have larger errors.

Figure 6 proves that the PB space has been clustered according to object dynamics. Neglecting the small portions of bouncing motions, the PB space is segmented into five areas. The rolling area intersects the sliding area, dividing it into two. This is due to the experimental condition using a single camera to observe object dynamics. Therefore, the model was incapable of distinguishing the rolling and sliding motions of the objects However, considering the rolling motion and sliding motion as a same dynamics, the model was capable of segregating the PB space into three areas based on the falling over, bouncing, and sliding/rolling motions.

To quantitatively analyze the PB space, the authors have compared the PB values shown in Fig. 5 with the clusters shown in Fig. 6. The number of PB values in the correct cluster and incorrect cluster are shown in Table IV. Since the model was incapable of distinguishing the sliding motion and rolling motion, 3 of the 6 errors for the sliding motion existed in the rolling cluster. The bouncing motion and rolling motion are similar since a small bouncing height of the ball resembles a rolling motion. These have created errors for the bouncing and rolling motions, both existing in the boundary between the bouncing and rolling clusters. 4 of the PB values for errors in falling over motions existed in the boundary of the falling cluster and sliding cluster. The boundary represents both falling over and sliding motions as described before. The rest of the errors (3 for fall over and 3 for slide) resulted from training errors of RNNPB and effects of robot motion during self-organization.

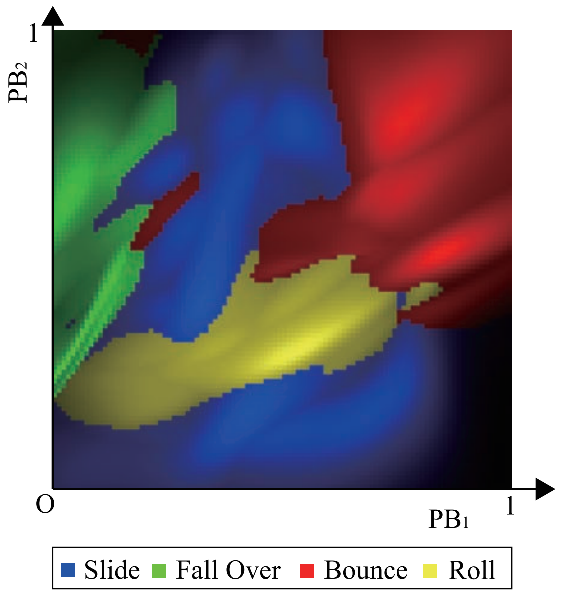


Fig. 6. Analysis of PB Space

TABLE IV
TRAINED PB VALUES AND CLUSTERS

	Correct Cluster	Incorrect Cluster
Slide	18	6
Fall Over	2	7
Bounce	12	1
Roll	10	3

C. Object Feature Sequence

Examples of the training results for the sliding, falling over, bouncing, and rolling motions are shown in Fig. 7. The other sequences had the same characteristics as those shown in Fig. 7. Open loop output is the sequence created by inputting the actual data acquired during active sensing. Closed loop output is the sequence created by *association*, where the output of the previous step is input at the current step. The value sequences of each of the four input nodes for the object feature of RNNPB is presented. The extracted sequence, open loop output, and closed loop output trace nearly the same trajectory. The smallness of the closed loop error denotes that the model possesses the capability to predict the object dynamics from the robot motion and initial object image.

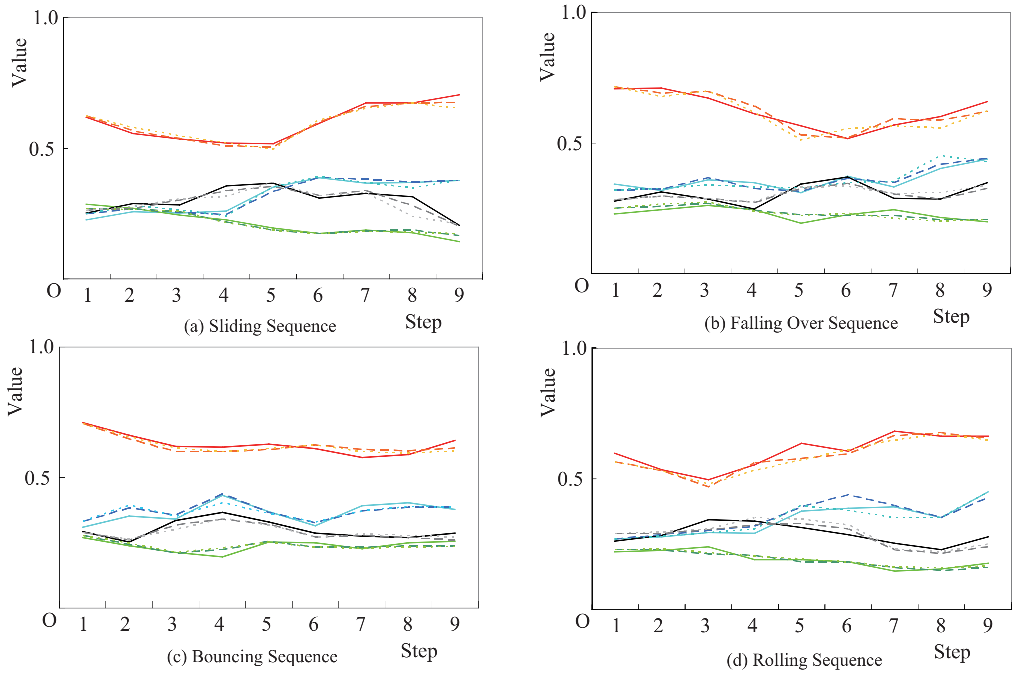
From Fig. 7, it is notable that each sequence possesses a different characteristic. Node 3 for the falling over sequence decreases, where it increases for the sliding sequence and rolling sequence. Due to the characteristics of the sliding and rolling motions (they both move perpendicular to the longer principal axis of inertia), similar types of features are extracted. Focusing on Node 1, there is a unique characteristic for the bouncing sequence. The value of node 1 fluctuates resembling the actual bouncing motion of a ball. These nodes do not represent well-known dynamics-representing features (such as center position), since they are not the principal features that discriminate the dynamics. For example, since the robot pushes the object right, every object moves right. Although the results do not provide a concrete description of the object dynamics, they prove to be sufficient to distinguish the difference of the dynamics.

V. Discussions

In this section, we present some discussions considering the experimental results.

A. Formation of PB Space

From Fig. 5, it is notable that some PB values exist in the incorrect area (cluster). As the input/output of RNNPB is the robot motor value and the object feature values, the PB space is self-organized according to the similarities of the two sequences. In this experiment, we decided the number of PB nodes as two, since it would be easier to analyze the results. However, the results have shown that two PB nodes are insufficient for clustering based only on object dynamics, neglecting the robot motion. Therefore, PB values existing in incorrect areas are those that were drawn due to the robot motion sequences represented by the transition of the shoulder pitch angle. To obtain a better prediction



	Extracted Sequence (Outputs of Hierarchical NN)	Open Loop Output	Closed Loop Output
Node 1	—	– –	·····
Node 2	—	– –	·····
Node 3	—	– –	·····
Node 4	—	– –	·····

Fig. 7. Extracted Feature Sequences and Training Results

capability, a larger number of PB nodes would be necessary to cluster the object dynamics.

It is also notable that a larger number of PB representing sliding motion overlap with the rolling motion compared to other object motions. This is due to the experimental condition that only a single camera is used as a sensor. Therefore, it is difficult for the model to distinguish the difference between the sliding motion and rolling motion of the object (the difference exists only in the speed of the object motion). In order to distinguish the two motions, it would be necessary to introduce additional sensors such as force sensors, as sliding objects would continuously input signals while rolling objects would input just an impulse signal into the sensor.

Although the results have shown some incorrect clustering, they imply the capability of the model to extract dynamic object features that describe the dynamics.

B. Model Configuration and Feature Extraction

In this paper, the authors have decided the configuration of the neural networks empirically. This configuration affects greatly the training result as it affects the number of local minimums.

In the experiment, the initial weights of the neural networks were decided randomly within a certain boundary. The boundary, [-0.7, 0.7], was also decided empirically. With a smaller boundary, the extracted sequences become closer to a constant value. This results in failure for the training process, as no object features will be extracted since every sequence would become similar. Thus, the PB space would be clustered only by the similarities of the motor sequences. With a larger boundary, the sequences fluctuate, compelling difficulties for the training process to converge. The authors have decided the boundary by investigating the initial fluctuation of the sequences.

Deciding the number of nodes for the neural networks is also a difficult task for the training process to converge. As preliminary experiments, the authors have trained the model with fewer nodes. Although clusters of each object dynamics were created, a larger number of PB values were distributed in wrong clusters. As described in Section III-B, the model should not possess unnecessarily large number of nodes as they would create additional local minimums.

The feature extraction hierarchical neural network is used for easing the training process of the model. Training RNNPB with raw image sequences without the feature extraction model would consume large amounts of time. Setting the configuration (number of nodes) of RNNPB would also become a difficult problem for such system.

Future works involve decision of the most appropriate configuration of the model, based on the image sequences. This would provide robustness to the training process of the model to extract the most appropriate features for describing the dynamics of the objects.

C. Dynamic Object Feature and Transformational Invariants

Transformational invariants are flows of dynamical object features, which are used to perceive motion. The simplest example of a transformational invariant is the flow of the center position of an object. In this case, the dynamical features to be extracted by our model would be the sequence of the center position.

The repertoire of invariants a person possesses, depends greatly on one's experience. A newly born baby has nearly no invariants whereas a mature child would have enough invariants to perceive almost any phenomenon in the world. In this experiment, the robot was trained using only the pushing motion with its left arm. Although the motion sequences vary depending on the shape of the objects, they have a same characteristic that they all move from the left to right. Therefore, the extracted object features do not contain a clear representation of the motion of the center position. The self-organized result of the PB space, however, proves that the model has extracted dynamic features describing the dynamics of the objects. The motion sequences presented in Fig. 7 also express an abstract description of object dynamics. By increasing the number of motion patterns, the robot would acquire more comprehensible features of the objects. This would involve improvement of the training capability of RNNPB to learn a larger variety of motion patterns, which is also left for future work.

VI. Conclusions

In this paper, the authors proposed a method to autonomously extract dynamical object features based on the robot's active sensing experiences. The training model consists of a sequential learning module, namely RNNPB, and a hierarchical neural network for the feature extraction module, linking the object feature input of RNNPB and the output of the hierarchical neural network. Using the training data acquired during active sensing, a simultaneous learning method is applied to train the two modules.

Experiments were conducted by active sensing twenty objects with the pushing motion of a humanoid robot Robovie II-s. The model was trained using 59 patterns of image and motor sequences. The background of the images were eliminated using color information. From the results, the model was capable of extracting features to distinguish the falling over, bouncing, and sliding/rolling motions, segregating the PB space into three areas. Therefore, the model is capable of extracting dynamic features which represent transformational invariants, based on the robot's active sensing experiences.

Future works involve resolution of the following issues.

1) Evaluation with additional sensors.
2) Evaluation of motion generation capability.
3) Introduce a criteria to decide the model configuration.

The first issue would be necessary for integrating multiple sensory data. As described in Section V-A, this would provide additional capabilities in distinguishing the dynamics of objects. The second issue is a prerequisite for modeling affordance. As features representing invariants are extracted from the model, the next step would be to link them to motion generation. The authors plan to take *Reliable Predictability* into consideration for motion generation, as done in our previous work [6]. The third issue would develop the model to general purposes. A difficulty in creating the model is decision of the optimal number of nodes and initial weights. These affect local minimums which induce difficulties to the training process. A concrete criteria for configuring the model would lead to a robust model. We believe that these works would lead to understanding the mechanism of affordance based on constructive approach.

VII. Acknowledgments

This research was partially supported by Global COE, the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Scientific Research (S), Grant-in-Aid for Young Scientists (A), Grant-in-Aid for Exploratory Research, RIKEN, and Kayamori Foundation of Informational Science Advancement.

References

[1] J. J. Gibson, "The Ecological Approach to Visual Perception," *Houghton Mifflin*, ISBN: 0898599598, 1979.

[2] D. Kim, J. Sie, S. M. Oh, J. M. Rehg, and A. F. Bobick, "Traversability Classification using Unsupervised On-line Visual Learning for Outdoor Robot Navigation," in *Proc. ICRA*, pp. 518-525, 2006.

[3] E. Uğur, M. R. Doğar, M. Çakmak, and E. Şahin, "The learning and use of traversability affordance using range images on a mobile robot," in *Proc. ICRA*, pp. 1721-1726, 2007.

[4] R. Bajcsy, "Active Perception," in *IEEE Proc., Special issue on Computer Vision*, Vol. 76, No. 8, pp. 996-1005, 1988.

[5] A. Stoytchev, "Behavior-Grounded Representation of Tool Affordances," in *Proc. ICRA*, pp. 3060-3065, 2005.

[6] S. Nishide, et al., "Object Dynamics Prediction and Motion Generation based on Reliable Predictability," in *Proc. ICRA*, pp. 1608-1614, 2008.

[7] P. Fitzpatrick and G. Metta, "Grounding Vision Through Experimental Manipulation," in *Philosophical Transactions of the Royal Society: Mathematical, Physical, and Engineering Sciences*, 361:1811, pp. 2165-2185, 2003.

[8] G. Metta and P. Fitzpatrick, "Early Integration of Vision and Manipulation," in *Adaptive Behavior special issue on Epigenetic Robotics*, Vol. 11, Issue 2, pp. 109-128, 2003.

[9] R. E. Shaw, M. McIntyre, and W. Mace, "The Role of Symmetry in Event Perception," in R. MacLeod and H. Pick Jr. (Eds.), "Studies in Perception: Essays in honor of J. J. Gibson," *Cornell University Press*, pp. 276-310, 1974.

[10] S. Nishide, T. Ogata, J. Tani, K. Komatani, and H. G. Okuno, "Predicting Object Dynamics from Visual Images through Active Sensing Experiences," *Advanced Robotics*, Vol. 22, No. 5, pp. 527-546, 2008.

[11] J. Tani and M. Ito, "Self-Organization of Behavioral Primitives as Multiple Attractor Dynamics: A Robot Experiment," *IEEE Trans. on SMC Part A*, Vol. 33, No. 4, pp. 481-488. 2003.

[12] M. Jordan, "Attractor dynamics and parallelism in a connectionist sequential machine," *Eighth Annual Conf. of the Cognitive Science Society* (Erlbaum, Hillsdale, NJ), pp. 513-546, 1986.

[13] D. Rumelhart, G. Hinton, and R. Williams, "Learning internal representation by error propagation," in *D. E. Rumelhart and J. L. McLelland, editors Parallel Distributed Processing (Cambridge, MA: MIT Press)*, 1986.

[14] R. Yokoya, T. Ogata, J. Tani, K. Komatani, H. G. Okuno, "Experience Based Imitation Using RNNPB," *Advanced Robotics*, Vol. 21, No. 12, pp. 1351-1367, 2007.

[15] H. Ishiguro, et al., "Robovie: an interactive humanoid robot," *Int. Journal of Industrial Robotics*, Vol. 28, No. 6, pp. 498-503, 2001.

[16] T. Ogata, S. Matsumoto, J. Tani, K. Komatani, and H. G. Okuno, "Human-Robot Cooperation using Quasi-symbols Generated by RNNPB Model," in *Proc. IEEE ICRA*, pp. 2156-2161, 2007.